Mục lục

Ký hiệu

Chúng tôi cung cấp một mô tả ngắn gọn về các ký hiệu sẽ được dùng thống nhất trong cuốn sách này. Chúng tôi sẽ mô tả hầu hết các khái niệm toán học dưới đây trong các chương 2 - 4.

Số và Mảng (array)

a	Đại lượng vô hướng (kiểu nguyên hoặc kiểu thực)
$\boldsymbol{a}$	Vector
$\boldsymbol{A}$	Ma trận
$\mathbf{A}$	Tensor
$\boldsymbol{I}_n$	Ma trận đơn vị n hàng n cột
$\boldsymbol{I}$	Ma trận đơn vị với số chiều tùy thuộc ngữ cảnh
$\boldsymbol{e}^{(i)}$	Vector cơ sở chính tắc $[0, \ldots, 0, 1, 0, \ldots, 0]$ với 1 ở vị trí i
$\text{diag}(\boldsymbol{a})$	Ma trận đường chéo với các phần tử của đường chéo chính là $\boldsymbol{a}$
a	Biến ngẫu nhiên dạng vô hướng
$\mathbf{a}$	Biến ngẫu nhiên dạng vector
$\mathbf{A}$	Biến ngẫu nhiên dạng ma trận

Tập hợp và đồ thị

$\mathbb{A}$	Tập hợp
$\mathbb{R}$	Tập hợp số thực
$\{0, 1\}$	Tập hợp bao gồm 0 và 1
$\{0, 1, \ldots, n\}$	Tập hợp bao gồm tất cả các số nguyên giữa 0 và n
$[a, b]$	Khoảng số thực bao gồm cả a và b
$(a, b]$	Khoảng số thực không bao gồm a nhưng bao gồm b
$\mathbb{A} \backslash \mathbb{B}$	Tập hợp trừ, nghĩa là tập hợp bao gồm các phần tử thuộc $\mathbb{A}$ mà không thuộc $\mathbb{B}$
$\mathcal{G}$	Đồ thị
$Pa_{\mathcal{G}}(\mathrm{x}_i)$	Nút cha của x_i trong đồ thị $\mathcal{G}$

Đánh chỉ số

a_i	Phần tử thứ i của vector $\boldsymbol{a}$, được đánh chỉ số từ 1
a_{-i}	Tất cả các phần tử của vector $\boldsymbol{a}$ trừ phần tử thứ i
$A_{i,j}$	Phần tử ở hàng thứ i cột thứ j của ma trận $\boldsymbol{A}$
$\boldsymbol{A}_{i,:}$	Hàng thứ i của ma trận $\boldsymbol{A}$
$\boldsymbol{A}_{:,i}$	Cột thứ i của ma trận $\boldsymbol{A}$
$\mathsf{A}_{i,j,k}$	Phần tử ở vị trí (i, j, k) của một tensor 3-D $\mathbf{A}$
$\mathbf{A}_{:,:,i}$	Lát cắt 2−D của một tensor 3−D
a_i	Phần tử thứ i của một vector ngẫu nhiên $\mathbf{a}$

Toán tử đại số tuyến tính

$\boldsymbol{A}^\top$	Chuyển vị của ma trận $\boldsymbol{A}$
$\boldsymbol{A}^+$	Giả nghịch đảo Moore-Penrose của $\boldsymbol{A}$
$\boldsymbol{A} \odot \boldsymbol{B}$	Tích theo từng phần tử (Hadamard) của $\boldsymbol{A}$ và $\boldsymbol{B}$
$\det(\boldsymbol{A})$	Định thức của $\boldsymbol{A}$

Phương pháp tính (calculus)

$\frac{dy}{dx}$	Đạo hàm của y theo x
$\frac{\partial y}{\partial x}$	Đạo hàm riêng của y theo x
$\nabla_{\boldsymbol{x}} y$	Gradient của y theo $\boldsymbol{x}$
$\nabla_{\boldsymbol{X}} y$	Ma trận đạo hàm của y theo $\boldsymbol{X}$
$\nabla_{\mathbf{X}} y$	Tensor chứa đạo hàm của y theo $\mathbf{X}$
$\frac{\partial f}{\partial \boldsymbol{x}}$	Ma trận Jacob $\boldsymbol{J} \in \mathbb{R}^{m \times n}$ của $f : \mathbb{R}^n \leftarrow \mathbb{R}^m$
$\nabla^2_{\boldsymbol{x}} f(\boldsymbol{x})$ hoặc $\boldsymbol{H}(f)(\boldsymbol{x})$	Ma trận Hesse của f tại điểm đầu vào $\boldsymbol{x}$
$\int f(\boldsymbol{x}) d(\boldsymbol{x})$	Tích phân xác định trên toàn bộ miền của $\boldsymbol{x}$
$\int_{\mathbb{S}} f(\boldsymbol{x}) d(\boldsymbol{x})$	Tích phân xác định của $\boldsymbol{x}$ trên tập hợp $\mathbb{S}$

Lý thuyết xác suất và lý thuyết thông tin

$\mathrm{a} \perp \mathrm{b}$	Hai biến ngẫu nhiên a và b độc lập
$\mathrm{a} \perp \mathrm{b} \mid \mathrm{c}$	Hai biến ngẫu nhiên a và b độc lập có điều kiện khi biết c
$P(\mathrm{a})$	Phân phối xác suất của một biến rời rạc
$p(\mathrm{a})$	Phân phối xác suất của một biến liên tục hoặc một biến có kiểu chưa xác định
$\mathrm{a} \sim P$	Biến ngẫu nhiên a có phân phối P
$\mathbb{E}_{\mathrm{x}\sim P}[f(x)]$ hoặc $\mathbb{E}f(x)$	Kỳ vọng của $f(x)$ theo $P(\mathrm{x})$
$\mathrm{Var}(f(x))$	Phương sai của $f(x)$ theo $P(\mathrm{x})$
$\mathrm{Cov}(f(x), g(x))$	Hiệp phương sai của $f(x)$ và $g(x)$ theo $P(\mathrm{x})$
$H(\mathrm{x})$	Entropy Shannon của biến ngẫu nhiên x
$D_{\mathrm{KL}}(P\|Q)$	Độ phân kỳ Kullback-Leibler của P và Q
$\mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$	Phân phối Gausian của $\boldsymbol{x}$ với trung bình $\boldsymbol{\mu}$ và hiệp phương sai $\boldsymbol{\Sigma}$

Hàm số

$f : \mathbb{A} \leftarrow \mathbb{B}$	Hàm số f với miền xác định $\mathbb{A}$ và miền giá trị $\mathbb{B}$
$f \circ g$	Phép hợp của hai hàm f và g
$f(\boldsymbol{x}; \boldsymbol{\theta})$	Hàm số của $\boldsymbol{x}$ được tham số hoá bởi $\boldsymbol{\theta}$. (Đôi khi chúng tôi viết là $f(\boldsymbol{x})$ và bỏ qua đối số $\boldsymbol{\theta}$ để tinh giản ký hiệu)
$\log x$	Logarit tự nhiên của x
$\sigma(x)$	Hàm logit sigmoid, $\frac{1}{1+\exp(-x)}$
$\zeta(x)$	Hàm softplus, $\log(1 + \exp(x))$
$\|\boldsymbol{x}\|_p$	Chuẩn L^p của $\boldsymbol{x}$
$\|\boldsymbol{x}\|$	Chuẩn L^2 của $\boldsymbol{x}$
x^+	Phần dương của x, tức là $\max(0, x)$
$\mathbf{1}_{\mathrm{condition}}$	trả về 1 nếu thoả mãn điều kiện, ngược lại trả về 0

Đôi khi, chúng ta sử dụng hàm f có đối số vô hướng, nhưng lại áp dụng nó cho một vector, một ma trận hoặc một tensor: $f(\boldsymbol{x}), f(\boldsymbol{X})$ hoặc $f(\mathsf{X})$. Điều này biểu thị việc áp dụng hàm f theo từng phần tử của mảng. Ví dụ if $\mathsf{C} = \sigma(\mathsf{X})$, then $\mathsf{C}_{i,j,k} = \sigma(\mathsf{X}_{i,j,k})$ đối với tất cả các giá trị hợp lệ của i, j và k.

Tập dữ liệu và phân phối

P_{data}	Phân phối sinh dữ liệu
$\hat{p}_{\text{data}}$	Phân phối thực nghiệm, định nghĩa bởi tập huấn luyện
$\boldsymbol{x}^{(i)}$	Mẫu thứ i (đầu vào) của một tập dữ liệu
$y^{(i)}$ hoặc $\boldsymbol{y}^{(i)}$	Nhãn tương ứng với $\boldsymbol{x}^{(i)}$ trong bài toán học có giám sát
$\boldsymbol{X}$	Ma trận $m \times n$, với mẫu đầu vào $\boldsymbol{x}^{(i)}$ nằm ở hàng $\boldsymbol{X}_{i,:}$

Chương 1

Giới thiệu

ND: *Những phần có ghi* ***ND*** *là chú giải thêm của người dịch*

Từ xa xưa, con người đã có ước mơ tạo ra những cỗ máy có khả năng suy nghĩ. Khao khát này đã có từ thời Hy Lạp cổ đại. Những hình tượng thần thoại như Pygmalion, Daedalus, Hephaestus là những nhà phát minh vĩ đại, và Galatea, Talos, Pandora có thể được xem là các thực thể sống nhân tạo ([OM04]; [Spa96]; [Tan97])

Ngay từ khi những chiếc máy tính đầu tiên có khả năng lập trình được tạo ra, người ta đã tự hỏi xem liệu chúng có thể trở nên thông minh hay không, cho đến khi một cỗ máy thông minh như kỳ vọng ra đời sau đó một thế kỷ ([Lov42]). Ngày nay, trí tuệ nhân tạo (*artificial intelligence* - AI) đã trở thành một lĩnh vực với vô vàn ứng dụng thực tiễn và trở thành chủ đề thu hút rất nhiều các đề tài nghiên cứu trên toàn thế giới. Chúng ta sử dụng phần mềm thông minh để tự động hóa các công việc chân tay, nhận dạng hình ảnh, âm thanh, chẩn đoán y học và hỗ trợ nghiên cứu khoa học cơ bản.

Trong những ngày đầu của trí tuệ nhân tạo, ngành này đã nhanh chóng giải quyết những vấn đề tuy phức tạp đối với con người, nhưng lại tương đối đơn giản đối với máy tính - những vấn đề có thể diễn tả được bằng một danh sách những quy luật dưới dạng ngôn ngữ toán học chuẩn tắc. Tuy nhiên, thách thức thực sự đối với trí tuệ nhân tạo là giải quyết những tác vụ dễ thực hiện với con người nhưng khó diễn tả một cách chuẩn tắc - những vấn đề mà con người chúng ta xử lý một cách rất tự nhiên bằng trực giác, chẳng hạn như nhận dạng tiếng nói hoặc nhận dạng khuôn mặt.

Nội dung cuốn sách này xoay quanh một giải pháp cho những vấn đề ấy. Giải pháp này cho phép máy tính tự học từ những kinh nghiệm thu được và hiểu hơn về thế giới thông qua một hệ phân cấp các khái niệm, trong đó mỗi *khái niệm* (concept) được định nghĩa theo mối quan hệ của nó với những khái niệm đơn giản hơn. Bằng cách để máy tính tự động thu thập tri thức từ kinh nghiệm, cách tiếp cận này giúp giảm bớt gánh nặng cho con người trong việc mô tả các tri thức cần thiết cho máy tính một cách tường minh. *Hệ phân cấp khái niệm* cho phép máy tính học những khái niệm phức tạp từ những khái niệm đơn giản hơn. Nếu chúng ta vẽ một đồ thị miêu tả hệ phân cấp khái niệm này, thì đồ thị đó sẽ có rất nhiều lớp, và rất *sâu*. Vì vậy, ta gọi cách tiếp cận này là *học sâu* (deep learning).

Rất nhiều thành công ban đầu của AI ra đời trong phòng thí nghiệm, một môi trường không đòi hỏi máy tính phải có nhiều hiểu biết về thế giới bên ngoài. Ví dụ, hệ thống chơi cờ vua *Deep Blue* của công ty IBM đã đánh bại nhà vô địch thế giới Garry Kasparov vào năm 1997 ([Hsu02]). Cờ vua dĩ nhiên là một môi trường đơn giản, chỉ bao gồm 64 ô vuông và 32 quân cờ, trong đó mỗi quân cờ chỉ có thể di chuyển theo những quy tắc

xác định. Dù sáng tạo ra một chiến thuật chơi cờ hiệu quả là một thành tựu lớn, nhưng thách thức không nằm ở việc dạy cho máy tính hiểu về các quân cờ và cách di chuyển của chúng. Luật chơi cờ vua được biểu diễn đầy đủ bằng một số quy tắc và có thể lập trình một cách dễ dàng.

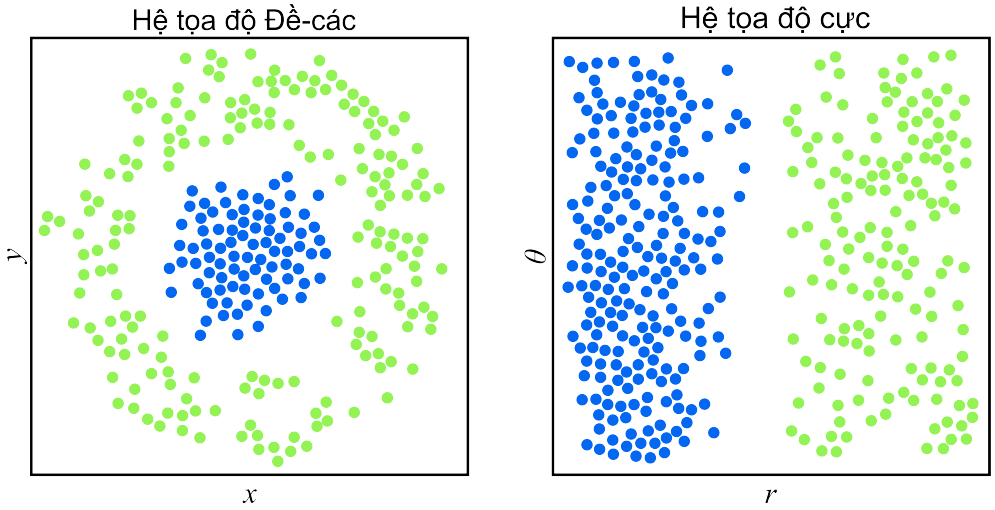
Trớ trêu thay, những tác vụ trừu tượng và chuẩn tắc khiến con người gặp trở ngại lại là những tác vụ dễ nhất với máy tính. Máy tính từ lâu đã có khả năng đánh bại người chơi cờ vua giỏi nhất, nhưng mới chỉ theo kịp trình độ của người bình thường trong việc nhận dạng vật thể và tiếng nói trong thời gian gần đây. Cuộc sống hàng ngày của một con người đòi hỏi một lượng tri thức khổng lồ về thế giới. Phần lớn những tri thức này mang tính chủ quan và phụ thuộc vào trực giác của mỗi người, nên rất khó để có thể diễn đạt chúng một cách tường minh. Máy tính cần nắm bắt được những tri thức như vậy về thế giới để có thể trở nên thông minh. Đưa những tri thức không tường minh vào trong một chiếc máy tính là một trong những thách thức chính trong ngành trí tuệ nhân tạo.

Một số dự án AI cố gắng gắn cứng tri thức về thế giới thông qua ngôn ngữ chuẩn tắc, để máy tính có thể tư duy về các chỉ lệnh trong ngôn ngữ chuẩn tắc thông qua các quy tắc suy luận logic. Phương pháp này được gọi là tiếp cận thông qua *cơ sở tri thức* (knowledge base). Tuy nhiên, không một dự án nào trong số đó đạt được thành công lớn. Một trong số những dự án nổi tiếng nhất là Cyc ([LG89]). Cyc là một công cụ suy luận và là một cơ sở dữ liệu gồm các chỉ lệnh được viết bằng ngôn ngữ CycL. Những chỉ lệnh này được đội ngũ nhân viên giám sát của Cyc đưa vào. Đó là một quá trình đầy gian nan. Người ta gặp khó khăn trong việc tạo ra những quy tắc tỉ mỉ, rất phức tạp để cố gắng miêu tả thế giới một cách chính xác nhất. Ví dụ, Cyc không hiểu được câu chuyện về một người tên Fred đang cạo râu vào buổi sáng ([Lin92]). Công cụ suy luận của Cyc đã gặp phải một chi tiết mâu thuẫn trong câu chuyện: nó biết rằng con người không có những bộ phận điện tử trên cơ thể, nhưng vì lúc đó Fred đang cầm một chiếc máy cạo râu, cho nên nó cho rằng thực thể mang tên "FredWhileShaving" có chứa những bộ phận điện tử. Cho nên Cyc đặt ra câu hỏi liệu Fred có còn là con người khi anh ta đang cạo râu hay không?

Những khó khăn mà các mô hình tri thức gắn cứng gặp phải cho thấy một hệ thống trí tuệ nhân tạo thực thụ cần có khả năng tự thu thập tri thức, bằng việc trích xuất các *mô thức* (pattern) từ dữ liệu thô. Khả năng này được gọi là *học máy* (machine learning). Sự xuất hiện của *học máy* cho phép máy tính giải quyết các vấn đề cần đến tri thức về thế giới thực và đưa ra quyết định chủ quan. Một thuật toán *học máy* đơn giản như *hồi quy logit* (logistic regression) có thể chẩn đoán và khuyến nghị có nên thực hiện phẫu thuật để hỗ trợ sản phụ khi sinh hay không ([Mor+90]). Một thuật toán khác, gọi là *giản luận Bayes* (naive Bayes), có thể phân biệt email thông thường và email rác.

Hiệu năng của những thuật toán học máy đơn giản này phụ thuộc nhiều vào *biểu diễn* (representation) của dữ liệu đầu vào. Ví dụ, khi sử dụng hồi quy logit để chẩn đoán khả năng cần phẫu thuật khi sinh, hệ thống AI không kiểm tra bệnh nhân một cách trực tiếp. Thay vào đó, các bác sĩ nạp một vài thông tin liên quan vào hệ thống, ví dụ như có tồn tại sẹo tử cung hay không. Mỗi mảnh thông tin liên quan đến bệnh nhân được gọi là một *đặc trưng* (feature). Hồi quy logit sẽ học mối tương quan giữa mỗi đặc trưng với nhiều kết quả đầu ra. Tuy nhiên, thuật toán này không thể quyết định việc các đặc trưng được định nghĩa thế nào. Nếu hồi quy logit được cung cấp một bản chụp MRI của bệnh nhân, thay vì báo cáo chi tiết của bác sĩ, nó sẽ không thể đưa ra dự đoán chính xác. Những điểm ảnh đơn lẻ trong bản chụp MRI không có nhiều tương quan với các biến chứng có thể xảy ra trong khi thực hiện phẫu thuật.

Sự phụ thuộc vào biểu diễn là hiện tượng phổ biến trong khoa học máy tính và cuộc sống hàng ngày. Trong khoa học máy tính, những thao tác như tìm kiếm trong một tập hợp dữ liệu có thể diễn ra nhanh hơn theo cấp số nhân nếu tập hợp đó được tổ chức có cấu trúc và được lập chỉ mục một cách thông minh. Đa phần chúng ta có thể dễ dàng thực hiện các phép toán số học trên hệ chữ số Ả Rập nhưng lại mất nhiều thời gian khi thực hiện trên hệ La Mã. Vậy nên không có gì ngạc nhiên khi việc lựa chọn cách biểu diễn có ảnh hưởng lớn đến hiệu suất của các thuật toán *học máy*. Hình 1.1 minh họa một ví dụ trực quan.



Hình 1.1: Ví dụ về sự khác biệt trong cách biểu diễn: giả sử chúng ta muốn phân loại hai dạng dữ liệu bằng cách vẽ một đường thẳng phân tách chúng trên *biểu đồ phân tán* (scatterplot). Trong hình bên trái, dữ liệu được biểu diễn theo hệ tọa độ Descartes (Đề-Các), và đường thẳng phân tách dữ liệu không tồn tại. Trong hình bên phải, ta biểu thị dữ liệu dưới hệ tọa độ cực và ta chỉ cần vẽ một đường thẳng dọc là có thể phân đôi được hai tập điểm. Hình được vẽ với sự giúp đỡ của David Warde-Farley.

Nhiều tác vụ trí tuệ nhân tạo có thể được giải quyết bằng cách thiết kế những đặc trưng phù hợp, rồi cung cấp bộ đặc trưng đó cho một thuật toán học máy đơn giản. Ví dụ, một đặc trưng hữu ích cho tác vụ nhận dạng người nói từ một đoạn âm thanh là kích cỡ thanh quản ước lượng của người đó. Đặc trưng này cho phép ta nhận biết được người nói là nam giới, nữ giới hay một đứa trẻ.

Tuy nhiên, trong nhiều tác vụ, rất khó để biết được những đặc trưng nào cần trích xuất. Ví dụ, chúng ta muốn viết một chương trình phát hiện xe hơi trong ảnh. Biết rằng xe hơi có bánh, nên ta cho sự xuất hiện của bánh xe là một đặc trưng. Tuy nhiên, rất khó để miêu tả chính xác hình dạng của một chiếc bánh xe thông qua giá trị của các điểm ảnh. Một chiếc bánh xe có hình dạng rất đơn giản, nhưng ảnh chụp của nó có thể trở nên phức tạp hơn nhiều do bóng đổ lên bánh xe, ánh nắng chói lóa ở những bộ phận kim loại của bánh xe, tấm chắn bùn của xe hoặc một vật thể nào đó che khuất vài bộ phận của bánh xe, và nhiều thứ khác nữa.

Một giải pháp cho vấn đề này là sử dụng học máy để không chỉ tìm được phép ánh xạ từ biểu diễn tới đầu ra, mà còn khám phá ra phép ánh xạ từ biểu diễn tới một biểu diễn khác. Hướng tiếp cận này được gọi là *học biểu diễn* (representation learning). Những biểu diễn học được thường hiệu quả hơn biểu diễn được thiết kế thủ công. Chúng cũng cho phép các hệ thống AI thích nghi nhanh với các tác vụ mới với rất ít sự can thiệp từ con

người. Một thuật toán học biểu diễn có thể tìm ra bộ đặc trưng phù hợp cho những tác vụ đơn giản trong vài phút, còn với những tác vụ phức tạp, nó cũng chỉ cần vài giờ tới vài tháng. Việc thiết kế thủ công các đặc trưng cho một tác vụ phức tạp yêu cầu nhiều thời gian và nỗ lực của con người, có thể khiến cả một cộng đồng đông đảo các nhà nghiên cứu tốn hàng thập kỷ để thực hiện.

Một ví dụ kinh điển của thuật toán học biểu diễn là *bộ tự mã hóa* (auto-encoder). Một *bộ tự mã hóa* là sự kết hợp của một *bộ mã hóa* (encoder), có chức năng biến đổi dữ liệu đầu vào thành một biểu diễn khác, và một *bộ giải mã* (decoder), có chức năng đưa biểu diễn mới trở về dạng ban đầu. Bộ tự mã hóa được huấn luyện để giữ lại nhiều thông tin nhất có thể khi dữ liệu đầu vào đi qua bộ mã hóa và bộ giải mã, nhưng chúng cũng được huấn luyện để các biểu diễn mới này có nhiều tính chất thú vị. Tồn tại nhiều loại bộ tự mã hóa khác nhau được thiết kế để đạt được những đặc tính mong muốn.

Khi thiết kế đặc trưng hay thiết kế thuật toán để học đặc trưng, mục tiêu của chúng ta là tách rời các *biến tố* (factors of variation) giúp giải thích dữ liệu quan sát được. Chữ **tố** ở đây, viết gọn của **nhân tố** (factor) ám chỉ những nguồn ảnh hưởng riêng lẻ. Những nhân tố như vậy thường là các đại lượng ta không quan sát được. Chúng có thể tồn tại dưới dạng các đối tượng không thể quan sát hay các lực không thể đo đạc trong thế giới vật chất, nhưng có tác động đến những đại lượng ta quan sát thấy. Chúng còn có thể tồn tại dưới dạng các thành tố trong tâm trí con người, giúp đưa ra cách giải thích đơn giản hơn hay suy diễn những nguyên nhân đằng sau dữ liệu thu thập được. Ta có thể coi chúng như những khái niệm hay những dạng trừu tượng giúp giải thích sự đa dạng trong dữ liệu. Ví dụ, khi phân tích một đoạn ghi âm lời nói, biến tố bao gồm tuổi, giới tính, chất giọng và những câu từ của người nói. Khi phân tích một bức ảnh chụp xe hơi, biến tố bao gồm vị trí, màu sắc, góc quan sát chiếc xe và độ sáng của ánh nắng mặt trời.

Một thách thức đối với nhiều ứng dụng AI trong thực tế đó là có nhiều biến tố ảnh hưởng tới mọi mảnh dữ liệu mà ta quan sát được. Những điểm ảnh trong bức ảnh chụp chiếc xe hơi màu đỏ có thể có màu gần với màu đen nếu ảnh được chụp vào ban đêm. Hình dạng cái bóng của chiếc xe tùy thuộc vào góc quan sát. Hầu hết các ứng dụng đòi hỏi việc *bóc tách* được các biến tố và loại bỏ những nhân tố không quan trọng khác.

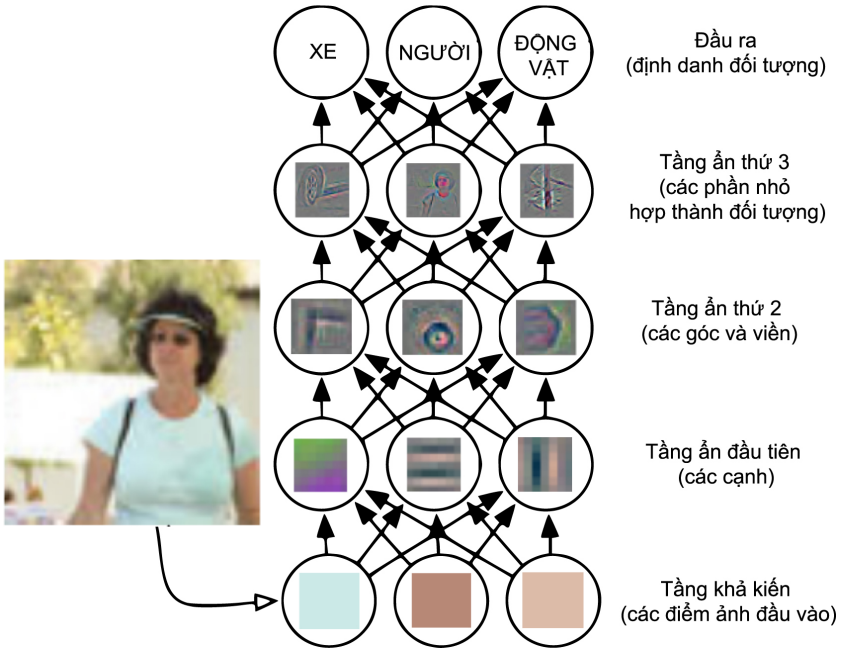
Dĩ nhiên, việc trích xuất các đặc trưng mức cao, trừu tượng từ dữ liệu thô có thể rất khó. Đa số những biến tố, chẳng hạn như chất giọng người nói, chỉ có thể được xác định khi AI có trình độ hiểu biết tinh tế, gần ngang trình độ của con người. Khi việc đạt được biểu diễn phù hợp có độ khó tương đương với giải quyết bài toán ban đầu, gần như học biểu diễn không giúp được gì nhiều cho chúng ta.

Học sâu (deep learning) giải quyết vấn đề trọng tâm này của học biểu diễn bằng cách đưa ra biểu diễn mới dựa trên những biểu diễn đơn giản hơn. Học sâu cho phép máy tính xây dựng các khái niệm phức tạp trên cơ sở là những khái niệm đơn giản hơn. Hình 1.2 cho thấy một hệ thống học sâu có thể biểu diễn khái niệm về một bức ảnh chụp con người bằng cách kết hợp các khái niệm đơn giản hơn, như các góc và viền, những khái niệm này lại được xác định từ khái niệm đơn giản hơn là các cạnh.

Ví dụ điển hình cho một mô hình học sâu là *mạng lan truyền thuận đa tầng* (feed forward deep network), hay thường được gọi là *mạng perceptron đa tầng* (multilayer perceptron - MLP). Một MLP thực chất là một hàm toán học ánh xạ dữ liệu đầu vào tới các giá trị đầu ra. Hàm này được hợp thành từ những hàm đơn giản hơn. Chúng ta có thể coi mỗi lần áp dụng một hàm toán học khác nhau là một lần đưa ra cách biểu diễn mới cho dữ liệu đầu vào.

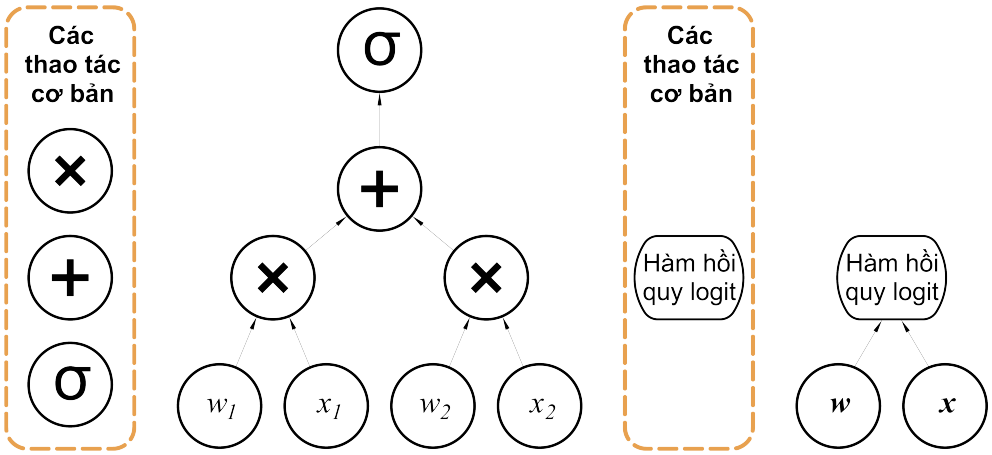
Ý tưởng về nhu cầu học biểu diễn đúng cho dữ liệu đã cho ta một góc nhìn về học

sâu. Một góc nhìn khác về học sâu đó là độ sâu của nó cho phép máy tính học ra một chương trình nhiều công đoạn. Mỗi tầng biểu diễn có thể được xem như trạng thái của bộ nhớ máy tính sau khi thực hiện một loạt các câu lệnh song song. Các mạng với độ sâu lớn hơn có thể thực hiện nhiều chuỗi lệnh hơn. Các chuỗi lệnh có công năng rất lớn, bởi những câu lệnh sau có thể sử dụng lại kết quả của những câu lệnh trước. Dưới góc nhìn này của học sâu, không nhất thiết rằng tất cả thông tin trong các *đơn vị kích hoạt* (activation) của một tầng đều mã hóa những biến tố giải thích dữ liệu đầu vào. Biểu diễn cũng lưu trữ các thông tin trạng thái giúp thực thi một chương trình có thể hiểu được dữ liệu đầu vào. Thông tin trạng thái này có thể tương đương với bộ đếm hay con trỏ trong một chương trình máy tính truyền thống. Dù không làm việc với nội dung cụ thể của đầu vào, nhưng nó giúp mô hình tổ chức quá trình xử lý.



Hình 1.2: Minh họa một mô hình học sâu. Thật khó để máy tính hiểu được ý nghĩa của các dữ liệu đầu vào ở dạng thô, chẳng hạn như bức ảnh này, nó được biểu diễn bằng một tập hợp các giá trị điểm ảnh. Ánh xạ từ tập các điểm ảnh đến định danh của vật thể là một hàm cực kỳ phức tạp. Trực tiếp học và đánh giá hàm ánh xạ này có vẻ như là nhiệm vụ bất khả thi. Học sâu giải quyết khó khăn này bằng cách chia nhỏ hàm ánh xạ phức tạp cần tìm thành một chuỗi các hàm ánh xạ đơn giản lồng vào nhau, mỗi ánh xạ được mô tả bởi một tầng khác nhau của mô hình. Đầu vào được đặt tại *tầng khả kiến* (visible layer), nó được đặt tên như vậy là bởi các biến số ở tầng này có thể quan sát được. Tiếp theo là một chuỗi các *tầng ẩn* (hidden layer) giúp trích xuất các đặc trưng có mức độ trừu tượng tăng dần. Chúng được gọi là "tầng ẩn" bởi giá trị ở những tầng này không có sẵn trong dữ liệu; thay vào đó, mô hình phải tự xác định những khái niệm nào là hữu ích trong việc lý giải mối liên hệ trong dữ liệu quan sát được. Những hình ảnh ở đây là hiển thị của các đặc trưng được biểu diễn ở mỗi tầng. Từ các điểm ảnh cho trước, tầng thứ nhất có thể dễ dàng nhận ra các cạnh biên, bằng cách so sánh độ sáng giữa các điểm ảnh lân cận với nhau. Từ các cạnh biên đã được mô tả bởi tầng ẩn thứ nhất, tầng ẩn thứ hai có thể dễ dàng tìm ra các góc và viền, có thể được nhận diện như tập hợp của các cạnh biên. Từ các góc và viền được mô tả bởi tầng ẩn thứ hai, tầng ẩn thứ ba có thể phát hiện ra toàn bộ thành phần của các vật thể cụ thể, bằng cách tìm các tập hợp đường bao và góc cụ thể. Cuối cùng, các mô tả dưới dạng các bộ phận của vật thể có thể được dùng để nhận diện vật thể trong ảnh. Những hình ảnh trên được tái hiện với sự cho phép của [ZF14].

Có hai cách thức chính để đánh giá độ sâu của một mô hình. Một là dựa trên chuỗi các lệnh cần thực hiện khi thực thi toàn bộ kiến trúc. Ta có thể xem nó như độ dài của đường đi dài nhất trong lưu đồ mô tả cách tính đầu ra của mỗi mô hình ứng với với đầu vào tương ứng. Cũng giống như việc hai chương trình máy tính tương đương sẽ có độ dài khác nhau phụ thuộc vào ngôn ngữ được sử dụng để viết chúng, cùng một hàm số có thể có lưu đồ với các độ sâu khác nhau phụ thuộc vào hàm chúng ta được phép sử dụng trong các bước đơn lẻ của lưu đồ. Hình 1.3 minh họa việc lựa chọn ngôn ngữ có thể dẫn tới độ sâu khác nhau cho cùng một kiến trúc như thế nào.



Hình 1.3: Minh họa một biểu đồ tính toán ánh xạ một đầu vào tới đầu ra, với mỗi node thực hiện một phép toán. Độ sâu là chiều dài của đường đi dài nhất từ đầu vào tới đầu ra, nhưng nó phụ thuộc vào định nghĩa thế nào là một bước tính toán. Biểu đồ trên minh họa quá trình tính toán của mô hình hồi quy logit, $\sigma(\boldsymbol{w}^\top \boldsymbol{x})$, trong đó σ là hàm sigmoid. Nếu ta sử dụng phép cộng, nhân và hàm sigmoid là các phép toán cơ bản trong ngôn ngữ lập trình, thì mô hình này có độ sâu là ba. Nếu ta coi hồi quy logit như là một phép toán cơ bản thì mô hình này chỉ có độ sâu là một.

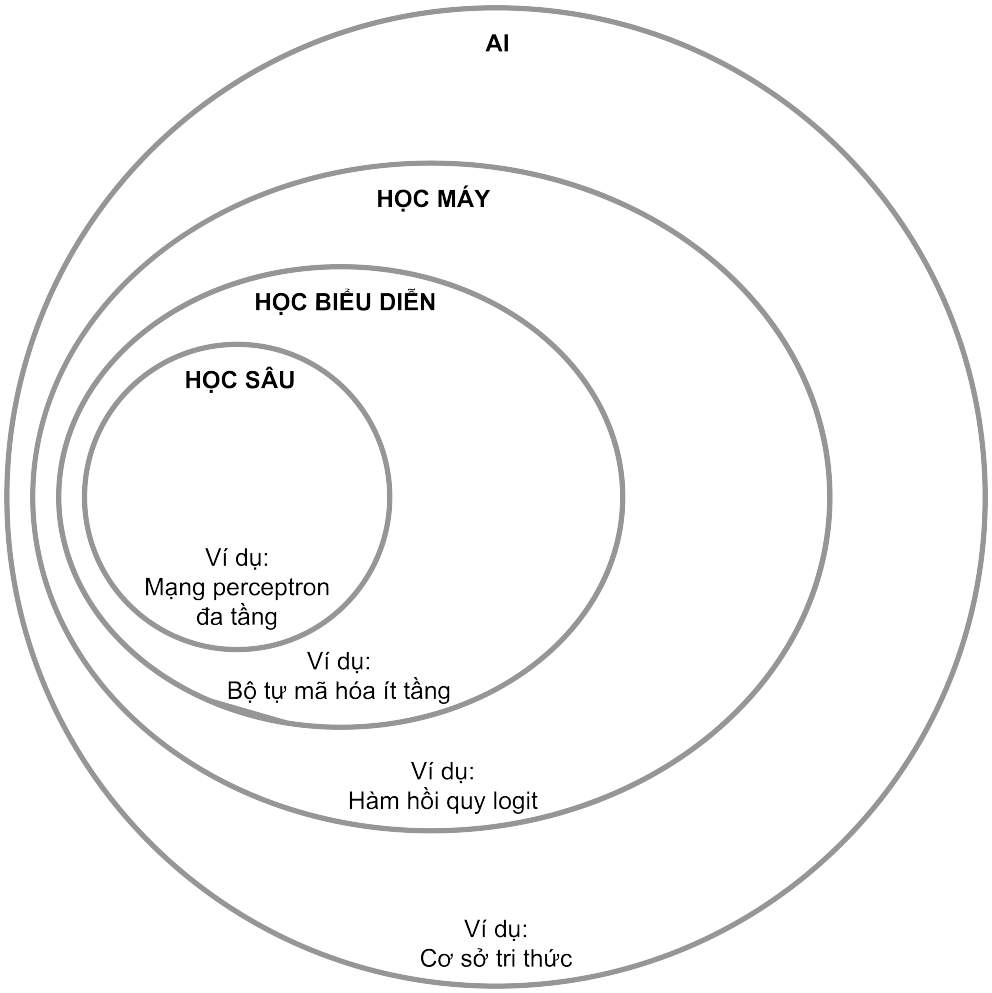
Cách thức xác định độ sâu thứ hai, sử dụng bởi các *mô hình đồ thị xác suất sâu* (deep graphical model), định nghĩa chiều sâu của một mô hình không phải là chiều sâu của lưu đồ tính toán, mà là của đồ thị mô tả liên hệ giữa các khái niệm với nhau. Trong trường hợp này, độ sâu của lưu đồ mô tả quá trình tính toán biểu diễn của mỗi khái niệm có thể lớn hơn nhiều so với đồ thị của chính khái niệm đó, bởi cách hiểu của hệ thống về các khái niệm đơn giản có thể được tinh chỉnh khi chúng ta có thêm thông tin về các khái niệm phức tạp hơn. Ví dụ, một hệ thống AI quan sát một hình ảnh của một khuôn mặt với một mắt trong bóng tối ban đầu sẽ chỉ nhìn thấy một mắt. Sau đó phát hiện rằng có một khuôn mặt, hệ thống có thể suy diễn rằng có thể tồn tại con mắt thứ hai trong ảnh. Trong trường hợp này, đồ thị của khái niệm chỉ bao gồm hai tầng, một tầng cho mắt và một tầng cho khuôn mặt, nhưng đồ thị tính toán lại bao gồm $2n$ tầng nếu chúng ta thực hiện tinh chỉnh ước lượng về mỗi khái niệm n lần khi biết các khái niệm khác.

Rất khó để xác định góc nhìn nào về độ sâu là phù hợp hơn - độ sâu của đồ thị tính toán hay độ sâu của mô hình đồ thị xác suất - và bởi vì mỗi người có cách chọn tập các thành phần cơ bản khác nhau để xây dựng đồ thị cho riêng mình, nên ta không có một giá trị độ sâu chính xác duy nhất cho một kiến trúc, tương tự với việc không có một giá trị duy nhất nào cho độ dài của một chương trình máy tính. Và chúng ta cũng không có quy ước về một mô hình như thế nào thì được xem là "sâu". Tuy vậy, chúng ta có thể coi học sâu là lĩnh vực nghiên cứu những mô hình bao gồm cả sự kết hợp một số lượng lớn

hơn các hàm hoặc các khái niệm học được, so với các mô hình học máy truyền thống.

Tóm lại, học sâu, chủ đề của cuốn sách này, là một hướng tiếp cận dành cho các bài toán AI. Cụ thể, nó là một dạng của học máy, một kỹ thuật cho phép hệ thống máy tính tự học từ trải nghiệm và dữ liệu. Chúng tôi dám chắc rằng học máy là hướng tiếp cận khả thi duy nhất để xây dựng các hệ thống AI có thể vận hành trong thế giới thực phức tạp. Học sâu là một dạng cụ thể của học máy, sở hữu sức mạnh và sự linh hoạt tuyệt vời thông qua việc học cách biểu diễn thế giới như một hệ phân cấp các khái niệm, trong đó mỗi khái niệm được định nghĩa từ những khái niệm đơn giản hơn, và mỗi biểu diễn được tính toán từ những biểu diễn kém trừu tượng hơn.

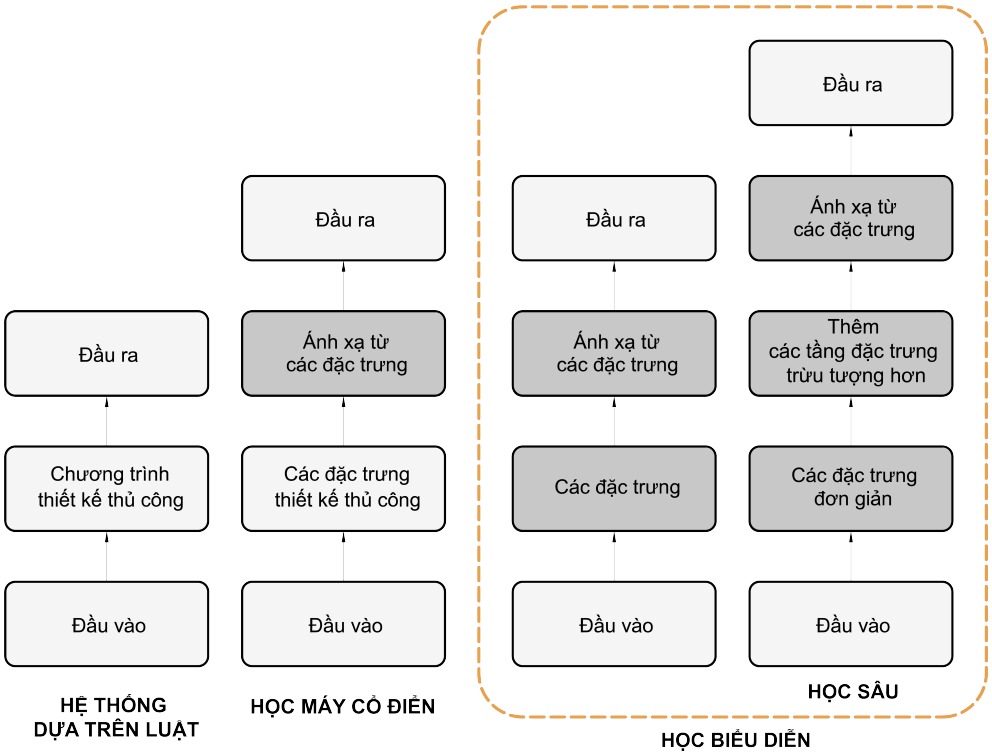
Hình 1.4 thể hiện mối liên hệ về phạm vi của các lĩnh vực khác nhau trong AI. Hình 1.5 mô tả một giản đồ về cách hoạt động của mỗi loại.



Hình 1.4: Biểu đồ Venn cho thấy học sâu là một dạng học biểu diễn, và học biểu diễn lại là một dạng học máy, một ngành được sử dụng trong nhiều (nhưng không phải toàn bộ) hướng tiếp cận AI. Mỗi phần của biểu đồ Venn bao gồm một ví dụ của một công nghệ AI.

1.1 Ai nên đọc cuốn sách này?

Cuốn sách có thể hữu ích với rất nhiều đối tượng độc giả, nhưng chúng tôi hướng tới hai đối tượng chính. Thứ nhất là sinh viên các trường đại học (hoặc sau đại học) đang



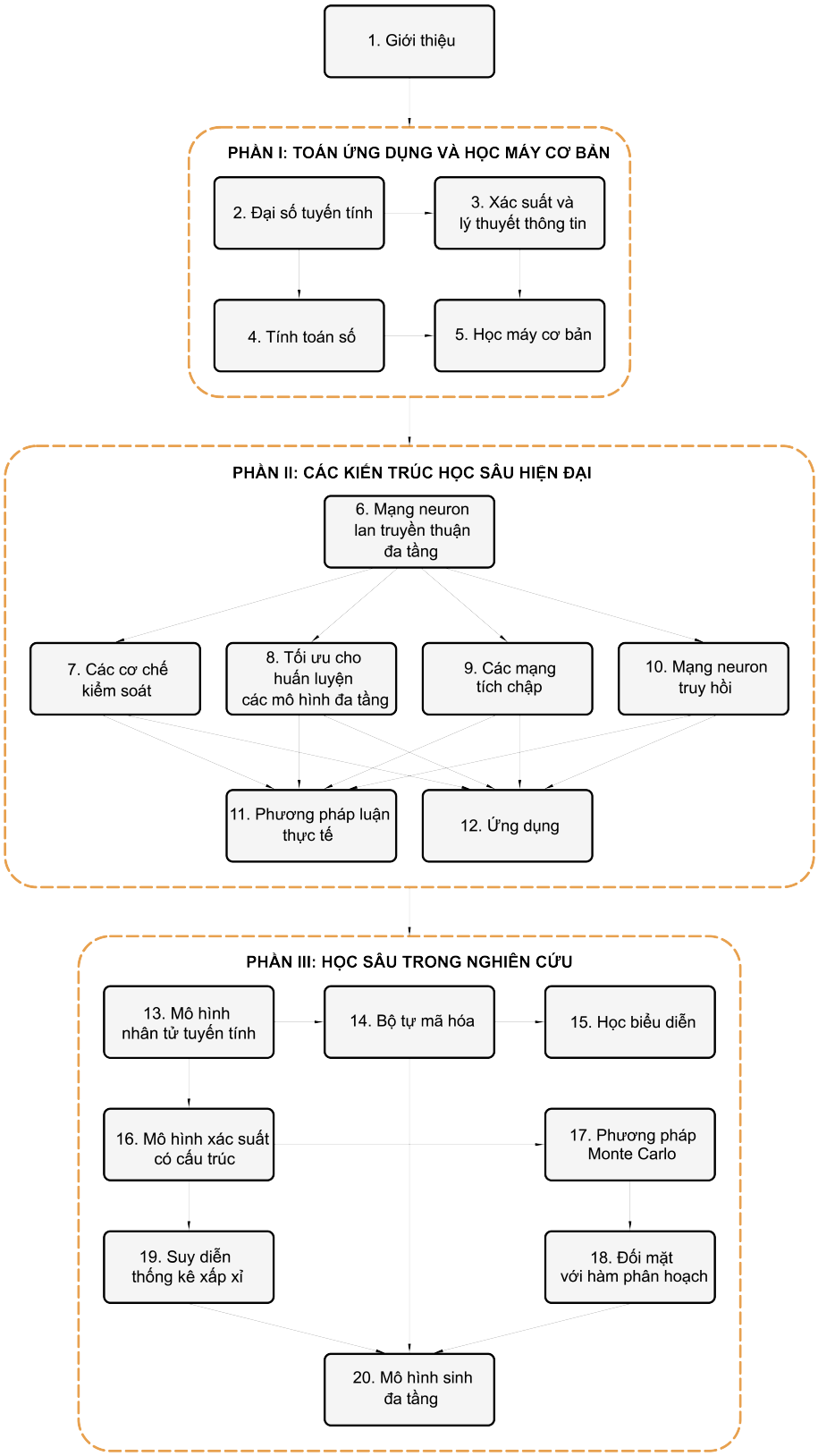
Hình 1.5: Biểu đồ cho thấy mối liên hệ giữa các thành phần khác nhau của một hệ thống AI trong mỗi hướng tiếp cận. Hình tô đậm là các thành phần có thể học được từ dữ liệu.

tìm hiểu về học máy, bao gồm những người đang bắt đầu sự nghiệp nghiên cứu trong lĩnh vực trí tuệ nhân tạo và học sâu. Thứ hai là các kỹ sư phần mềm, những người chưa có kiến thức nền tảng về học máy hay thống kê, nhưng muốn nhanh chóng tiếp thu kiến thức trong hai lĩnh vực này và bắt đầu sử dụng học sâu trong sản phẩm của mình. Học sâu đã chứng minh tính hữu dụng trong nhiều lĩnh vực phần mềm, bao gồm thị giác máy tính, xử lý âm thanh và tiếng nói, xử lý ngôn ngữ tự nhiên, robot, tin sinh học và hóa học, trò chơi điện tử, công cụ tìm kiếm, quảng cáo trực tuyến và tài chính.

Cuốn sách này được chia thành ba phần để giúp đỡ cho nhiều đối tượng đọc giả khác nhau một cách tốt nhất. Phần I giới thiệu các công cụ toán học cơ bản và những khái niệm học máy cơ bản. Phần II mô tả những thuật toán học sâu phổ biến và đã được phát triển một cách hoàn thiện nhất. Phần III mô tả những ý tưởng được suy đoán là quan trọng cho việc nghiên cứu học sâu trong tương lai.

Bạn có thể bỏ qua bất cứ phần nào trong sách nếu không quan tâm hoặc không phù hợp với kiến thức nền tảng của bạn. Ví dụ, độc giả quen thuộc với đại số tuyến tính, xác suất, và khái niệm học máy cơ bản có thể bỏ qua phần I, trong khi đó những người chỉ muốn áp dụng học máy vào sản phẩm sẽ không cần quan tâm tới phần III. Để giúp bạn lựa chọn chương cần đọc, chúng tôi cung cấp một lưu đồ tổng quan thể hiện bố cục nội dung trong sách.

Chúng tôi tạm giả sử rằng tất cả các bạn đọc đều có hiểu biết cơ bản về khoa học máy tính như: lập trình, hiểu biết cơ bản về các vấn đề hiệu năng tính toán, lý thuyết về độ phức tạp của thuật toán, *phương pháp tính* (calculus) cơ bản và một số thuật ngữ trong lý thuyết đồ thị.



Hình 1.6: Tổng quan bố cục nội dung cuốn sách. Mũi tên từ chương này tới chương khác có ý nghĩa rằng cần phải đọc chương xuất phát của mũi tên mới có thể hiểu được chương mà mũi tên chỉ đến.